**Software Requirements Specification (SRS)**

**Project Title**: Student Referral and Tier Reward Management System

**Prepared by**: Senior Systems Architect **Version**: 1.0 **Date**: April 1, 2025

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to outline the detailed software requirements for the development of a web-based referral and tier-based reward management system. The application allows former students to refer new candidates to courses and receive rewards based on referrals and conversions.

**1.2 Scope**

This system facilitates:

* Referral tracking
* Points allocation
* Tier-based benefits
* Admin management
* CSV import/export
* Secure, session-based user access

**1.3 Definitions, Acronyms, and Abbreviations**

* **SRS**: Software Requirements Specification
* **CSV**: Comma Separated Values
* **UI**: User Interface
* **I2CAN**: Institute managing the courses
* **SPA**: Single Page Application

**1.4 References**

* IEEE 830 SRS Standard

**1.5 Overview**

This document defines the functional and non-functional requirements, data storage approach, user workflows, and security constraints for the system.

**2. Overall Description**

**2.1 Product Perspective**

The system will be a standalone, multi-page web application integrated into an existing static website. All data is stored in JSON files.

**2.2 User Classes and Characteristics**

* **Admin**: Single user with complete system control
* **Referring Users**: Former students who can refer candidates

**2.3 Operating Environment**

* Web-based: HTML5, CSS, JavaScript frontend
* Backend: Python (no database)
* Data Storage: JSON files (users.json, referrals.json, logs.json)

**2.4 Design and Implementation Constraints**

* No third-party DBMS
* All authentication is managed through locally stored credentials

**2.5 User Documentation**

* Online help/manual for admin and user functionality

**2.6 Assumptions and Dependencies**

* Static website integration assumed
* Single admin user scenario

**3. Specific Requirements**

**3.1 Functional Requirements**

**3.1.1 User Authentication**

* Admin and users login via credential-based authentication.
* Auto logout after 15 minutes of inactivity.

**3.1.2 Referral Management**

* Users can refer candidates via a form or CSV upload.
* Fields: Name, City, Relationship (Friend, Relative, Colleague, Other), Qualification, Course Interested, Phone Number, Email (optional)
* Duplicate check based on name/email/phone.
* Statuses: Pending, Approved, Converted, Rejected.

**3.1.3 Points System**

* 50 points per referral
* 250 points for conversions (admin-verified)
* Additional 1% of admission amount (input by admin)
* Admin can add/remove points manually

**3.1.4 Tier Management**

* Tiers: Bronze (0), Silver (10,000), Gold (25,000), Diamond (50,000)
* Weekly recalculation (Sundays)
* Tier downgrade warnings 2 weeks prior
* Auto tier change with congratulatory message

**3.1.5 Admin Panel**

* Create/edit/delete users
* Reset passwords
* Mark referrals as converted or rejected
* Edit referral details
* View dashboard, generate reports
* Upload users via CSV with validation and row-level error feedback

**3.1.6 Reporting and Analytics**

* Downloadable reports: weekly, monthly, custom date range
* CSV format
* Sample CSV templates available for uploads
* Visual dashboards with charts

**3.1.7 Logging and Auditing**

* Actions logged in logs.json (timestamp, user, action)

**3.2 Non-Functional Requirements**

**3.2.1 Performance**

* Supports up to 100 active users and 500 referrals/month

**3.2.2 Security**

* No password hashing (initial version)
* Minimum password rules: 8+ characters, at least 1 uppercase, alphanumeric

**3.2.3 Usability**

* Responsive multi-page HTML UI
* User dashboard: points, referrals, tier, benefits, referral form/upload

**3.2.4 Portability**

* Designed for integration with existing static site

**4. External Interface Requirements**

**4.1 User Interfaces**

* **Login Page**
* **Dashboard** (points, tier, referral form, CSV upload)
* **Admin Panel** (user mgmt, referral mgmt, reporting)

**4.2 Hardware Interfaces**

* N/A

**4.3 Software Interfaces**

* Local file I/O for reading/writing JSON

**4.4 Communications Interfaces**

* Localhost or hosted via simple HTTP server for internal deployment

**5. Data Structures**

**5.1 users.json**

{

"users": [

{

"username": "user1",

"password": "Password123",

"name": "John Doe",

"tier": "Silver",

"points": 10500,

"referrals": ["ref\_001", "ref\_002"]

}

]

}

**5.2 referrals.json**

{

"referrals": [

{

"id": "ref\_001",

"referrer": "user1",

"name": "Jane Smith",

"phone": "9876543210",

"status": "Converted",

"admission\_amount": 50000,

"points\_awarded": 800

}

]

}

**5.3 logs.json**

{

"logs": [

{

"timestamp": "2025-04-01T10:00:00Z",

"user": "admin",

"action": "Added 250 points to user1 for ref\_001"

}

]

}

**6. Appendices**

**Appendix A: Tier Benefits Table**

| **Tier** | **Points Needed** | **Free Courses** | **Product Discount** | **Referral Bonus** |
| --- | --- | --- | --- | --- |
| Bronze | 0 | None | 0% | 5% |
| Silver | 10,000 | 1 | 2% | 5% |
| Gold | 25,000 | 2 | 4% | 6% |
| Diamond | 50,000 | All | 6% | 10% |

**Appendix B: Referral Status Definitions**

* **Pending**: Newly submitted, awaiting admin action
* **Approved**: Validated by admin
* **Converted**: Enrolled; points updated
* **Rejected**: Invalid or fake; points revoked

**End of Document**